PixVOD: Pixel-Distributed Direct Visual Odometry and Depth Estimation

Shinjeong Kim, Ignacio Alzugaray, Callum Rhodes, Paul H. J. Kelly, and Andrew J. Davison

Department of Computing, Imperial College London
{s.kim, i.alzugaray, c.rhodes, p.kelly, a.davison}@imperial.ac.uk

Abstract. Images composed of 2D pixel arrays are the standard input to computer vision algorithms, yet many underlying computations can be distributed across pixels. Transmitting raw, redundant, and noisy pixel data off the sensor remains inefficient, motivating a shift toward focal-plane sensor-processors that perform a significant part of the computation directly within each pixel. We envision pixels synthesizing higher-level signals locally, reducing downstream load, and providing richer inputs for higher-level vision tasks.
We propose a fully parallelizable form of visual odometry and depth estimation across pixels, where sensor-processors exchange information through Gaussian Belief Propagation (GBP) to achieve consensus about camera motion and infer depth from per-pixel photometric observations and a surface normal prior. To maintain geometric stability during optimization, we introduce a keyframe-like anchoring mechanism that regulates the effective baseline between frames, enabling consistent motion and depth updates. Our method is evaluated on realistic datasets, demonstrating the feasibility of GBP-based pixel-level distributed odometry and depth estimation with keyframe anchoring on-sensor. Project Page: https://www.shinjeongkim.com/pixvod/

Keywords: Visual Odometry · Camera Tracking · Gaussian Belief Propagation

1 Introduction

Mobile edge applications like robotics or wearables demand results from computer vision at high frame rates, whilst maintaining low latency, high accuracy, and especially with *minimal energy consumption* in battery-driven products. A promising and fundamental path to achieving all of these goals is to move away from the standard separation of camera and processor, connected by video transmission, towards hardware which combines visual sensing and processing in a single unit. This could process visual input locally, rapidly, without high bandwidth data flow, and output only the necessary high-level estimates for the application in question.

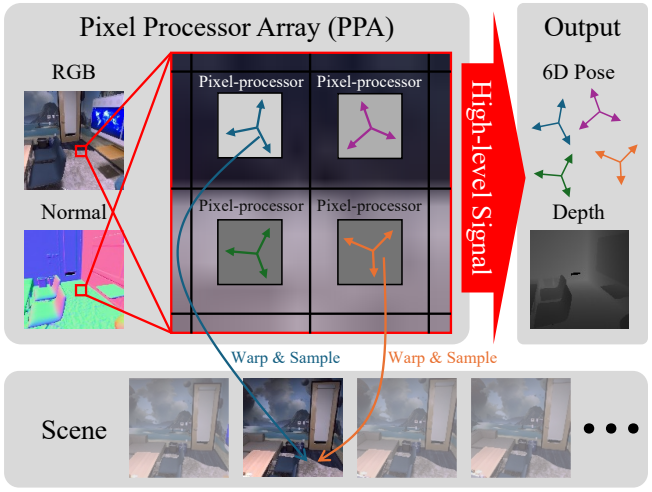

Fig. 1: Most vision algorithms assume the use of an off-sensor central processor with access to all pixel information. In contrast, we present an algorithm to track camera pose and estimate depth on the assumption that each pixel has its own local processing capability, with only local memory and the ability to exchange messages with other pixels. Our goal is to motivate and guide the design of near-future chips able to carry out fundamental early vision tasks on-sensor, with extremely high efficiency and speed.

At the limit, in the *On Sensor Vision* paradigm, processing can be built into a camera's image sensor itself. Prototypes already exist, such as the SCAMP series of vision chips [9, 12]. These implement a Pixel Processor Array (PPA) design, where each photo-sensitive pixel in a standard rectangular grid has a programmable local processing capability, a set of memory registers, and the ability to exchange information with neighbouring pixels. Crucially, a PPA achieves maximum performance when it uses *only* this local, pixelwise memory.

Despite the limited computational capacity of these chips, impressive image processing capabilities [8,32,49] have already been demonstrated, such as feature extraction and matching, local motion estimation, object tracking, and image classification and segmentation via a small CNN, with extremely high speed and low power usage. Research is now underway to design the next generation of such chips, with questions about how much pixel-wise computation and memory, and what communication structure will enable more capability.

In this paper, we focus on the fundamental vision competence of local 6DoF motion estimation from a single camera, often known as visual odometry (VO), and investigate how this could be achieved with pixel-parallel computation to be suitable for a near-future on-sensor vision device. Previous work harnessing SCAMP for VO and SLAM has performed feature extraction and even matching [8,32] on-sensor, but these methods have required frame-rate communication

with an external CPU to solve the optimisation needed for 6DoF motion estimation.

Here, we propose an algorithm to achieve 6DoF VO estimation with pure in-pixel computation. The key challenge here is how to achieve consistent estimates of camera motion, a global property, across the distributed memory structure of a PPA. We argue that a *dense* VO approach, where camera motion is estimated alongside a full dense depth map, is actually easier to achieve on a PPA than a sparse approach where geometry is estimated only for sparse points. This is because of the continuity of real scenes, which can be harnessed as priors on dense depth and therefore can be implemented as local pixel-wise constraints.

We formulate our approach as a distributed factor graph, stored across the whole pixel array with variables at every pixel to estimate camera motion and scene depth. We use identity factors to synchronize a global camera motion estimate across the whole array.

2 Related Work

There is a growing amount of work exploring vision algorithms parallelized for processors with local computation and memory. The SCAMP PPA vision chips have been used to enable high-speed and robust keypoint tracking and matching [8, 32], and this has been proven as the front-end to visual odometry, but with the use of an external processor for 3D estimation. Meanwhile, the same chips have also been shown to be capable of running small CNN and RNN networks for classification and action recognition [26, 49].

Other papers which are closest to our work have started to explore whether global geometric estimation can be achieved fully in-pixel. This work, like ours, was not yet implemented on true PPA chips due to the current limitations of hardware like SCAMP, which has a tiny amount of analogue and digital memory per pixels, can only perform simple arithmetic computations, and has a strict grid-wise pixel communication structure. Instead, it has aimed to explore the design space for near future chips with extra capabilities. Scona *et al.* [47] investigated 6DoF visual odometry for RGB-D sensors, with a model of pixel-parallel processing with both local and long-distance random communication structure implemented using a Graphcore IPU [27], while PixRO [2] studied in-pixel camera rotation tracking and compared the value of gridwise and hierarchical communication structured. Both of these papers adopted factor graph and GBP formulations similar to our own work, where we now tackle the full monocular 6DoF visual odometry and depth estimation problem.

As Structure from Motion (SfM) [1,25,46] has been practically utilized as an annotation method for 3D datasets, there have also been efforts to parallelize or accelerate its camera pose optimization. Theoretically-oriented approaches, such as rotation [13] and translation [39, 62] averaging, achieve faster convergence by carefully analyzing, reformulating, and/or relaxing the underlying optimization constraints. On the implementation side, [24,38] aims to accelerate computation by leveraging a deeper understanding of actual hardware architectures. However,

these methods focus on optimizing camera poses at the image level, rather than pursuing pixel-level parallelization.

In Visual SLAM, there have been several recent investigations of accelerations with specialized processing hardware [3, 7, 16, 45, 53]. Our focus on camera tracking and depth estimation is aligned with speeding up the tracking-and-mapping frontend [23, 28, 36, 59]. Nevertheless, most efforts in this vein parallelize only resource-intensive subroutines on specialized hardware, and the overall pipeline remains centralized.

It is well established that monocular depth estimation is highly dependent on the global context, modelled with globally connected Markov Random Fields [57] or Transformers [6, 54]. In contrast, surface normal estimation has been shown to achieve remarkable performance even with models that possess only local receptive fields [4], empirically demonstrating that surface normals can be effectively estimated from relatively local information compared to depth [43, 44]. This observation justifies our assumption of a per-frame normal prediction being available from a PPA.

Normal integration [10, 11, 19, 21] is the problem of reconstructing depth from a surface normal field, and in camera pose tracking and mapping contexts, numerous methods have explored its use to reconstruct depth or enhance its quality, or to be jointly optimized with other geometric objectives [29, 40, 60]. However, these approaches typically assume a conventional centralized off-sensor processor. In our work, we exploit the spatial sparsity inherent in the optimization formulation of the recent normal integration method BINI [10] to enable efficient pixel-level parallelization.

Probabilistic graphical models [5] and their associated optimization techniques [41, 42, 58] have been widely used long before the rise of deep learning. Even after deep learning–based approaches have become dominant, they have remained prevalent as post-processing tools for fine-grained refinement [22], as optimization back-ends for learning [17, 33, 56], and as powerful inference mechanisms in resource-constrained settings [34]. Recent theory further justifies Gaussian approximations for BP in sparse factor graphs [55]. In our work, we model the camera pose and depth estimation problem as a Gaussian factor graph [15] and employ Gaussian Belief Propagation to perform pixel-level distributed inference. The GBP formulation is naturally amenable to distributed optimization in many types of graph structure [31, 38].

3 Methodology

3.1 Direct Visual Tracking with Centralized Computation

Before explaining the full details of our method, for clarity and as a baseline for later comparison, we first explain the simpler standard formulation of direct visual 6DoF tracking against a rigid scene when computation and memory are centralized.

We consider a calibrated camera that undergoes general 6DoF motion in a static scene. Given $\mathcal{I}_s$ (the "source image", from time 0) and $\mathcal{I}_t$ (the "target

image", from time 1), we seek the relative pose $\mu_{[0:6]} \in \mathbb{SE}(3)$ and a source-view depth map $\exp(\mu_{[6]}) \in D \in \mathbb{R}_+^{H \times W}$ that jointly explain their photometric differences.

In high frame-rate tracking, inter-frame camera motion and visual changes are small and we can perform direct, iterative photometric alignment. Formally, we align the warped target image $\mathcal{I}_t$ and the source image $\mathcal{I}_s$, over the overlapping domain $\Omega(\mu)$ induced by the camera motion parameterized by μ. The resulting nonlinear problem is:

$$\mu^* = \arg\min_{\mu} \sum_{p \in \Omega(\mu)} \rho\left(\|\mathcal{I}_s[p] - \mathcal{I}_t[\mathcal{W}(p;\mu)]\|\right)^2, \tag{1}$$

where ρ denotes a robust cost function designed to effectively ignore pixels whose photometric residuals are unreliable due to large correspondence errors, thereby enhancing the robustness of the system.

As in [35,47], the warping function $\mathcal{W}$, geometrically maps each pixel location p in $\mathcal{I}_s$ and its corresponding depth $\exp(\mu_{[6]})$ to the corresponding location in $\mathcal{I}_t$:

$$\mathcal{W}(p;\mu) = \pi\left(K \exp(\mu_{[0:6]}) K^{-1} \pi^{-1}(p, \exp(\mu_{[6]}))\right) \ , \tag{2}$$

where the projection function is defined as $\pi(\mathbf{x}) = [x/z, y/z]^T$, and K is the camera intrinsic matrix. Note that here, the left multiplication of $\exp(\mu_{[0:6]})$ is considered as an application of rigid transformation.

When optimizing Eq. (1), if all pixel information is accessible at each iteration, an optimal update can be computed for the current linearized model. However, this assumes that the entire image is loaded into globally accessible memory, which corresponds to the traditional setting where images are transmitted off-sensor and processed by a centralized processor. We will refer to this as the centralized approach, and use it as a comparison later.

3.2 Distributed Formulation with a Gaussian Factor Graph

Our method is designed to implement the same direct visual tracking process but with computation and memory distributed across a pixel-parallel array of processing cores. We formulate the dense visual odometry and depth estimation problem as a Gaussian factor graph [15] describing the probabilistic relationship between variables of interest $v \in \mathcal{V}$ and factors $f \in \mathcal{F}$ which represent priors or measurements, such that the likelihood of the entire model is the product $p(\mathcal{V}) = \prod_i f_i(\mathcal{V}_{f_i})$, where $\mathcal{V}_{f_i} \subseteq \mathcal{V}$ is a subset of variables involved in each factor. This is the standard probabilistic formulation of a broad range of geometric estimation in SfM, SLAM, and robotics [15], and is the same formulation adopted by PixRO [2] and BP-SF [47].

The key to our distributed implementation is that the stacked variable associated with each pixel can be stored locally at that pixel, and information about the factors that connect pixel variables can be stored altogether at the pixels at either end of the connection. As we will explain later, we implement optimisation

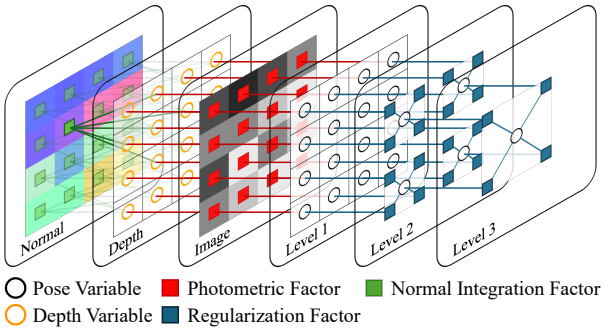

Fig. 2: Topology of the proposed factor graph. The factor graph on the right side of the Image layer, related to camera pose estimation, follows the sharded topology of PixRO [2], with the domain extended to $\mathbb{SE}(3)$. The factor graph on the left side of the Image layer reconstructs depth through the normal integration and photometric factors. For the normal integration factors, note that all factors except the one located at the first row, first column are rendered semi-transparent to better illustrate the connection structure, whose topology is otherwise less comprehensible. For visual clarity, the prior factors for both pose and depth variables are omitted from the diagram.

to achieve marginal estimates of the variables via Gaussian Belief Propagation, which operates via local computation and message passing. Therefore no global memory access is needed by the pixels.

The overall structure of our factor graph for one timestep of visual odometry and depth estimation is shown in Fig. 2. A full spatiotemporal solution to sequential VO would involve a temporal stack of such graph fragments per timestep, connected by appropriate continuity factors. Here, we concentrate on explaining one timestep, where the goal is to estimate the camera motion from one image frame to the next, along with a depth map for the first frame.

At each pixel we store a stacked variable representing 3D camera motion and depth. We model 3D camera motion and rotation using $\mathbb{SE}(3)$ pose variables. Each pixel stores its own $\mathbb{SE}(3)$ estimate, but since we know that there is only one global motion of the whole camera, these variables are connected by strong Gaussian regularisation "identity" factors which encourage them to converge to the same value. We have experimented with different patterns that these factors could take, and we will discuss this later. Fig. 2 shows a hierarchical, sharded structure as was shown to have good performance in PixRO [2], while being able to implement even on existing PPAs with a few hops of direct-neighbor communication.

Each pixel has a photometric factor, which models the "optical flow" constraint that the camera motion and depth of a pixel determine with which pixel it should line up in the subsequent image frame, and that this corresponding pixel should have the same intensity. Finally, we use factors that constrain the depth estimates of neighbouring pixels. These factors could simply model depth smoothness, but we have found it more effective to use factors that constrain

the depth of neighbouring pixels based on normal predictions coming from a local network, in the style of BINI [10], which showed that normal integration is highly suitable for pixel-level parallelization. We assume that accurate normal prediction from a small CNN is something that should be readily achievable in a pixel array in the near future, following work such as [26]. We will explain the details of this factor graph model shortly.

3.3 Detailed Description of Factors

In our distributed formulation, each pixel p_i stores a stacked variable μ_i for each pixel p_i, whose first six elements $\mu_{i,[0:6]}$ are a local estimate of global camera motion, and whose final element $\mu_{i,[6]}$ is an up to scale estimate of the log-depth of the scene at that pixel. Our factor graph model has the following factors which constrain the absolute and relative values of these variables.

Photometric factors: This factor corresponds to the decomposition of the original centralized formulation in Eq. (1) into local terms, each dependent on a single pixel, and represents the photometric difference between the corresponding pixel intensities of the two images:

$$E_D^i(\mu_i) = \frac{1}{2}\rho\left(\left\|\mathcal{I}_s\left[p_i\right] - \mathcal{I}_t\left[\mathcal{W}(p_i;\mu_i)\right]\right\|_{\Lambda_D}\right)^2 \tag{3}$$

This energy is computed as each pixel reads the photometric value at its corresponding warped location in the target image, assuming its own camera pose estimate as the target camera pose. On real PPA arrays, this can be implemented through local routing mechanisms that allow each processor to access neighboring pixel values.

Prior factors: This factor models a per-pixel prior hypothesis on the pose of the current camera. On the first frame of a sequence, the prior pose is set to identity $[\mathbf{I};\mathbf{0}]$, serving as an assumption that the camera motion is slow, which stabilizes the optimization process—particularly during early iterations.

At later frames in a high frame-rate image sequence, this factor can propagate motion continuity priors from one timestep to the next.

$$E_P^i(\mu_i) = \frac{1}{2}\left\|\mu_i \diamondminus \hat{\mu}_i\right\|_{\Lambda_P}^2 \tag{4}$$

We also have factors that constrain the overall scale of the log depth at each pixel. For the first frame in a sequence, we use a prior mean depth of 1, corresponding to a log-depth prior of 0. In subsequent frames, these factors can propagate depth estimates per pixel to future frames.

Camera motion identity factors: These are the crucial factors which encourage the multiple estimates of camera motion at each pixel towards a unique, global value. Without such factors, each pixel would find its own photoconsistent motion estimate, and in low-texture or repetitive scenes, these estimates would be highly inconsistent.

$$E_R^{i,j}(\mu_{i,[0:6]},\mu_{j,[0:6]}) = \frac{1}{2}\left\|\mu_{i,[0:6]} \boxminus \mu_{j[0:6]}\right\|_{\Lambda_R}^2 . \tag{5}$$

Since these factors are syncronizing a global variable, rather than having a local image interpretation, they can in principle be connected in any pattern. The simplest pattern would be to match the grid structure of pixels, with factors connecting all horizontally and vertically neighbouring pixel pairs. This is desirable from the point of view of matching the communication structure of current PPA hardware such as SCAMP, and we will continue to experiment further with this, but we currently find that it can require many iterations of GBP optimisation to achieve good global camera pose estimates with only these local connections. In the main experiments in this paper, we use a hierarchcal or sharded connection pattern for these identity factors.

Normal integration factors: Finally, we use factors which help us to estimate smooth and consistent dense log depth across all pixels of the source image, the components $\mu_{i,[6]}$ of each pixel's variable μ_i. Without such factors, which encode priors about the smoothness of the real world, accurate reconstruction would be possible only in extremely textured scene regions where photoconsistency alone might suffice.

Rather than assuming only that local scene depth changes smoothly, we use a stronger prior on the assumption that local estimates of surface orientation (normal map) are available. It is feasible to run small CNNs on a pixel-parallel array (e.g. [26, 49], and normal prediction can be performed extremely well with CNNs [4] as the scene normals are relatively local properties, rather than global quantities such as scene depth. Previous work such as BINI [10] has shown that local integration of a normal map from a CNN can produce accurate local depth estimates up to scale within regions without depth discontinuities.

Our factors constrain the depth difference between neighbouring pixels to be consistent with the locally measured normals:

$$\begin{aligned} E_{N_x}^{i>j}(\mu_{i,[6]}, \mu_{j,[6]}) &= \frac{1}{2} \left\| (\mu_{i,[6]} - \mu_{j,[6]}) + \frac{n_x}{\tilde{n}_z} \right\|_{\Lambda_N}^2 \\ E_{N_y}^{i>j}(\mu_{i,[6]}, \mu_{j,[6]}) &= \frac{1}{2} \left\| (\mu_{i,[6]} - \mu_{j,[6]}) + \frac{n_y}{\tilde{n}_z} \right\|_{\Lambda_N}^2 , \end{aligned} \tag{6}$$

where the upper equation applies to horizontally adjacent pixels and the lower one to vertically adjacent pixels. $\tilde{n}_z$ denotes the z-component of the normal scaled to the image-plane coordinate system (see p. 4 of BINI [10] for the exact definition), while n_x and n_y denote the x and y components of the surface normal, respectively.

3.4 Iterative Gaussian Belief Propagation

To achieve camera motion and depth estimation, we need to rapidly and incrementally optimise this factor graph model to obtain marginal estimates of the camera motion and depth variables. The desire to implement this on a pixel-parallel array rules out standard algorithms and software libraries which assume that all variables and factors are jointly accessible. Instead, we use Gaussian

Belief Propagation (GBP), which can obtain marginal estimates via fully distributed computation by passing Gaussian messages locally between factors and variables. Assuming Gaussian distributions, the messages admit analytic update rules, allowing nodes and factors to update locally without explicit synchronization. In the following, we summarize the steps tailored to the pose and depth of the camera; fuller details appear in [2, 14, 30, 31, 37].

In our method, each pixel stores a stacked variable $v \in \mathcal{V}$ which represents camera pose and log-depth in $\langle \mathbb{SE}(3), \mathbb{R} \rangle$. These variables are generally related through nonlinear factors; thus, the linearization point must be progressively approximated. Following the conventions in [2, 30, 31], we take the linearization point of each pose-log-depth variable to be its current mean, denoted by $\bar{\mu} \in \langle \mathbb{SE}(3), \mathbb{R} \rangle$. The corresponding probabilistic variable is then defined as:

$$v = \bar{\mu} \boxplus {}^{\bar{\mu}}\xi, \quad \text{where} \quad {}^{\bar{\mu}}\xi \sim \mathcal{N}(0, \Sigma_v) \ . \tag{7}$$

Here, ${}^{\bar{\mu}}\xi \in \langle \mathfrak{se}(3), \mathbb{R} \rangle$ denotes a stacked variable consisting of a Lie-algebra element and a scalar component. We regard this as an element of a Euclidean space, exploiting the vector-space structure of the Lie algebra to apply standard probabilistic methodologies. Throughout this work, we adopt the Lie-group notation of [50] for the direct product $\langle \mathbb{SE}(3), \mathbb{R} \rangle$: in particular, $\oplus$ denotes a pose update by composing with the exponential map, and $\boxplus$ denotes the componentwise extension of $\oplus$ to the composite manifold.

The residual of any factor involving such variables can then be approximated by a first-order Taylor expansion similar to [2] as follows:

$$\begin{aligned} r(\mu) = r(\bar{\mu} \boxplus \tau) &\approx r(\bar{\mu}) + \left.\frac{Dr}{D\mu}\right|_{\mu=\bar{\mu}} \tau \\ &= r(\bar{\mu}) + J(\bar{\mu})\tau = \bar{r} + \bar{J}\tau \ , \end{aligned} \tag{8}$$

where $J(\cdot)$ denotes the right Jacobian of $\langle \mathbb{SE}(3), \mathbb{R} \rangle$. Note that the Jacobian matrix is block diagonal, as this composite is a direct product group.

Then, each energy term can be accordingly approximated as follows:

$$\begin{aligned} E(\tau) = \frac{1}{2}||r(\bar{\mu} \boxplus \tau)||^2_{\Lambda_r} &\approx \frac{1}{2}||\bar{r} + \bar{J}\tau||^2_{\Lambda_r} \\ &\propto \frac{1}{2}\tau^T \bar{J}^T \Lambda_r \bar{J}\tau + (\bar{J}^T \Lambda_r \bar{r})^T \tau \ . \end{aligned} \tag{9}$$

We follow the standard steps of GBP to send factor-to-variable messages, variable-to-factor messages, to relinearize factors and to compute variable beliefs, as in [2, 30, 31].

3.5 Keyframe-based Tracking Strategy

Direct photometric tracking method is well known to have a limited basin of convergence around the optimum, and our method is likely to converge to suboptimal camera pose and depth when the inter-frame motion is large. In a PPA,

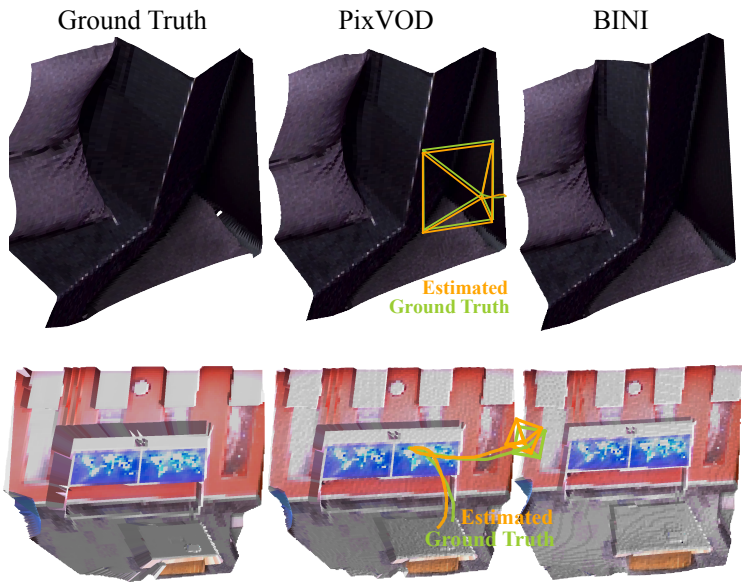

Fig. 3: Visualization of an estimated local trajectory and depth with **(top)** relatively small motion and being close to surface, **(bottom)** relatively large motion and being far from surface. The view frustum of the camera is visualized only on the final pose. For the top image, the trajectory from a reference keyframe to the next one is shown, and for the bottom trajectory, 5 keyframes are in between, while omitted for visibility of the trajectory. Unprojected depth was connected to form a mesh and textured with an image for easier comprehension. Note that the scale of the estimated depth of the proposed method is more accurately reconstructed compared to when BINI is solely used, as explained in Sec. 4.5.

on-board processing means there is no need to transmit video data and the frame-rate can be kept high so motions from one frame to the next can be kept small. However, we have found that simply running very high-rate frame-to-frame 6DoF odometry and naively integrating these estimates over time is also problematic: if the motion between the reference and target frames becomes excessively small, it becomes difficult to reliably disambiguate rotation and translation, degrading tracking accuracy. We therefore introduce a local keyframing strategy. Specifically, we treat reference frame as a fixed keyframe during tracking for a number of steps, while the target frame is updated incrementally as camera motion accumulates, and 6DoF poses are always estimated locally relative to this keyframe. In the following sections, we analyse the characteristic advantages and limitations of this keyframe-based formulation.

4 Experimental Evaluation

4.1 Implementation Details

As in most publicly available GBP implementations [14, 37], we alternate synchronously between the message passing stage and the belief update stage, iterating over both factor and variable nodes. Only the variable and factor-to-variable messages are stored explicitly. At each relinearization step, these variables and messages are projected onto the tangent space of the manifold at the updated linearization point. We employ the Huber function as a robust loss, which is a common choice in photometric optimization, and use the covariance weighting method from [14] in every GBP factor message passing step.

Our current implementation is a GPU simulation of a potential future implementation on a true pixel parallel array. All steps of the GBP algorithm are parallelized using JAX, and the experiments were conducted on a desktop machine equipped with an NVIDIA RTX 4090 GPU and an AMD Ryzen 9950X CPU.

4.2 Evaluation Details

For the proposed experiments, we generate sequences of images of 128 by 128 pixels from the publicly available Replica dataset [51], which provides photorealistic 3D reconstructions of real-world environments. We use the sequences *office_{0, 1, 2, 3}* adopted from iMap [52] and Nice-SLAM [61], but upsample each frame-to-frame interval by a factor of 10 to simulate the small baselines that would arise at high frame-rate in a PPA. For trajectory upsampling, we employ ScLERP [20], an extension of the linear interpolation method SLERP [48] from $\mathbb{SO}(3)$ to $\mathbb{SE}(3)$. Unless otherwise specified, ground-truth surface normals are used. All metrics on translation and depth are evaluated after the scale of estimation is aligned with that of ground-truth, by mimizing L_2 error.

Our hyperparameters are $\sigma_D, \sigma_P, \sigma_R, \sigma_N$, and t_{huber}, which are respectively used in the form $\Lambda = \frac{1}{\sigma^2} I$ and as a Huber threshold, and can be interpreted as controlling the assumed noise level of the underlying model (or simply, strength) of each factor. We empirically determine suitable magnitudes for these hyperparameters, and, unless otherwise specified, we set $(\sigma_D, \sigma_P, \sigma_R, \sigma_N, t_{\text{huber}}) = (5 \times 10^{-3}, 1, 4 \times 10^{-4}, 10^{-3}, 400)$ at the leaf level of our factor graph. The values of σ_P and σ_R are halved at each higher level of the sharded quadtree factor graph in Fig. 2 compared to the level below. This choice reflects the observation that lower levels have access only to more local information and therefore produce pose estimates that are less reliable than the globally aggregated poses at higher levels. We empirically validate this design choice in the supplementary material.

4.3 Analysis on Keyframe-based Tracking

In this section, we analyze the choice of keeping a reference frame and optimizing with respect to a continually updated target frame. Along the upsampled

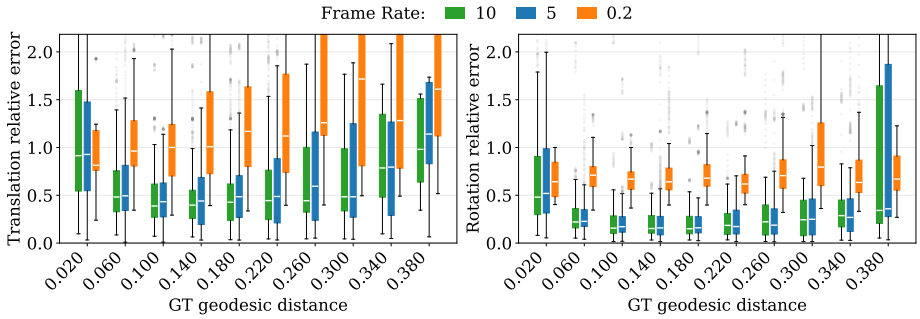

Fig. 4: Comparison of estimated pose error across various frame rates, based on the deviations of the target camera pose from origin $[\mathbf{I}; \mathbf{0}]$. As the pose change between consecutive frames increases, i.e., as the frame rate decreases, the convergence basin of the proposed method (with respect to the deviation of the target camera pose from the origin) becomes smaller, and the corresponding error increases. However, the benefit of reduced frame-to-frame motion exhibits diminishing returns (cf. framerates **10** and **5**), similar to which has also been observed in related domains such as event-based vision [18].

Replica trajectory, we sampled 10 keyframes per scene. In each keyframe, we extract ground-truth surface normals and use them as the prior for normal integration (Eq. (6)); for the subsequent 300 frames, we treat each as a target frame from which observations for the photometric factor (Eq. (3)) are sampled. For every target frame, we perform 100 synchronized GBP iterations, where one synchronized iteration applies message passing and belief update sequentially to all factors and variables. The choice of 100 iterations is the minimal count that suffices to reach a convergent solution for almost all target frames. At the end of the optimization loop for each target frame, the pose estimations of all pixels are averaged to form the final estimation for that frame. When moving on to the next frame, all variables are initialized with the solution from the previous frame.

Here, we analyze why keyframe–frame optimization is essential by varying the update frequency of target frames to enlarge inter-frame visual change, and comparing against the original update rate. (Fig. 4) Specifically, we follow the protocol mentioned above (framerate 10); we update the target frame every two frames while doubling the number of optimization iterations per target frame (framerate 5); and we update the target frame every fifty frames while performing fifty times as many iterations per target frame (framerate 0.2). Using the converged solution at the end of each target-frame optimization as a data point, we obtain the box-and-whisker plot in Fig. 4. The x-axis reports the geodesic distance of ground-truth, defined as the l_2-norm of the $\mathfrak{se}(3)$ vector of the relative pose of ground-truth with respect to the pose of the keyframe; the y-axis shows the relative error of our estimate: $||t_{\text{est}} - t_{\text{gt}}||_2 / ||t_{\text{gt}}||_2$ for translation (where t is taken from translation part of $\mathbb{SE}(3)$), and the axis-angle vector for rotation.

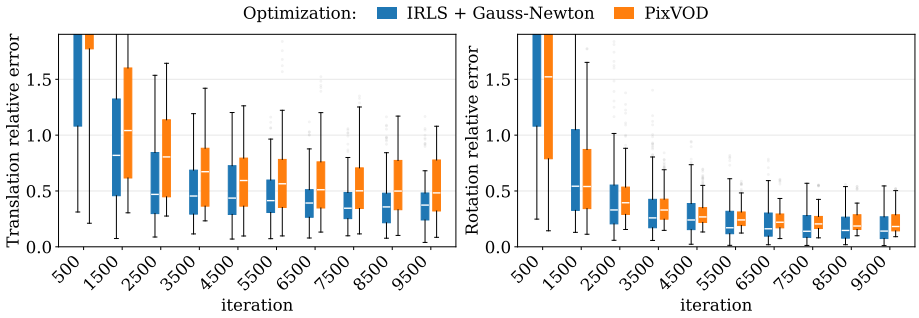

Fig. 5: Comparison of the proposed method against the baseline method, which is solved in a centralized manner. We formulated the centralized optimization problem as an Iterative Reweighted Least Squares (IRLS), and solved it with the Gauss-Newton method.

As shown in Fig. 4, for framerate 0.2 the method settles more frequently at suboptimal solutions as the target-frame updates become infrequent. However, comparing framerate 10 to 5 indicates that the benefit of further increasing the update frequency is limited; in real-world settings, this limit would be accentuated by the reduced signal-to-noise ratio caused by shorter exposure times. The plot also offers insight into when camera-pose tracking is particularly challenging: when motion is too small, it becomes difficult to disambiguate the motion type (e.g. vertical translation versus rotation about a horizontal axis), whereas when motion is too large, the region providing usable visual cues diminishes, making accurate pose tracking more difficult.

4.4 Comparison of Pixel-Parallel and Centralized Visual Odometry Estimation

We compare the optimization performance of the proposed GBP-based method against the aforementioned centralized baseline. The centralized approach (Sec. 3.1) optimizes a global estimator of pose and depth with simultaneous access to all pixel measurements, and is solved using Gauss–Newton. Since the problem is nonlinear as proposed in Eq. (1) and [10], we model it via iteratively reweighted least squares (IRLS), equally weighting two objectives. Following the earlier experiments, we set the first frame as the reference and treat the subsequent 100 frames, over which convergence occurs in the vast majority of cases statistically, as continuously updated target frames; we perform 100 iterations of optimization per frame in sequence.

As shown in Fig. 5, the centralized method converges faster and to higher accuracy than our approach, as expected; however, this presumes per-iteration access to global pixel information. In contrast, the proposed method accesses pixels only locally and independently at each variable, offering an alternative

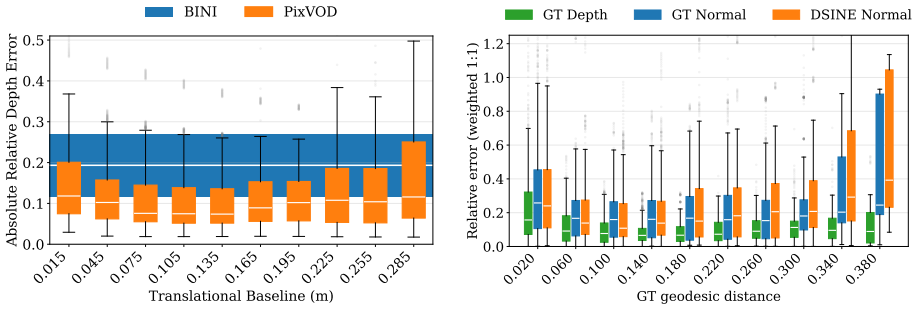

Fig. 6: **(a)** Comparison of reconstructed depth of BINI [10] after convergence against our reconstructed depth with various translational baselines of camera. **(b)** Evaluation on camera poses estimated with various structural priors. Note that most of the experiments reported in this paper are paired with the Ground Truth Normals.

optimization strategy in environments where global access is atypically expensive.

4.5 Photometrically Guided Normal Integration

In this section, we compare our method against pure normal integration without photometric guidance. In line with recent work [29, 60], the results indicate that incorporating photometric correspondences can extend normal-integration-based reconstruction beyond the object level, by guiding both sides of depth-discontinuous boundary to be pulled apart.

As shown in Fig. 6a and exemplified in Fig. 3, when initialized with the depth at BINI's convergence, our approach achieves substantially higher depth quality than BINI after its own optimization. Note that whiskers are omitted for the BINI results. Consistent with Sec. 4.3, there exists a range of camera translations for which the depth reconstruction gains are most pronounced.

4.6 Ablation Study on Geometric Prior

In Fig. 6b, we evaluate the proposed method with a variety of structural priors to verify that its pose estimation remains robust irrespective of the prior type. As expected, using ground-truth depth yields the best convergence. Interestingly, when using DSINE [4] normals, the quality of the converged pose is not substantially worse than with ground-truth normals, aside from a reduced basin of convergence. This indicates that our approach generalizes to real-world use when paired with practical normal-estimation methods.

5 Conclusions

We present the first per-pixel distributable algorithm for jointly optimizing camera pose and depth. Our approach demonstrates that each pixel-processor can infer global image properties using only its own pixel measurement, a local geometric prior, and message passing with neighboring processors. We also provide an effective strategy for camera tracking in the typical operating regime of PPAs, high frame rates with small inter-frame changes, and analyze the operating range and limitations of the method, providing insights that can inform the design of next-generation PPAs.

In the future, we will investigate whether gradual weakening of the inter-pixel factors that enforce a single global camera motion estimate, and so to what extent this will enable accurate non-rigid 3D scene flow estimation. We also plan to continue to experiment with different patterns of inter-pixel factor connections, to find the best compromise between patterns that enable rapid optimisation and those which are feasible and efficient to design into real future chips.

Acknowledgements

This research is supported by the EPSRC grant EP/Y020499/1. We appreciate all members of the Robot Vision Group for insightful discussions.

References

1. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building rome in a day. Communications of the ACM **54**(10), 105–112 (2011)
2. Alzugaray, I., Murai, R., Davison, A.: Pixro: Pixel-distributed rotational odometry with gaussian belief propagation. arXiv preprint arXiv:2406.09726 (2024)
3. Asgari, B., Hadidi, R., Ghaleshahi, N.S., Kim, H.: Pisces: power-aware implementation of slam by customizing efficient sparse algebra. In: 2020 57th ACM/IEEE Design Automation Conference (DAC). pp. 1–6. IEEE (2020)
4. Bae, G., Davison, A.J.: Rethinking inductive biases for surface normal estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 9535–9545 (2024)
5. Bishop, C.M., Nasrabadi, N.M.: Pattern recognition and machine learning, vol. 4. Springer (2006)
6. Bochkovskiy, A., Delaunoy, A., Germain, H., Santos, M., Zhou, Y., Richter, S., Koltun, V.: Depth pro: Sharp monocular metric depth in less than a second. In: Yue, Y., Garg, A., Peng, N., Sha, F., Yu, R. (eds.) Int. Conf. Learn. Represent. vol. 2025, pp. 75602–75637 (2025), https://proceedings.iclr.cc/paper_files/paper/2025/file/bc8b2058fd96978a4146f18298cb2d39-Paper-Conference.pdf
7. Boikos, K., Bouganis, C.S.: Semi-dense slam on an fpga soc. In: 2016 26th International Conference on Field Programmable Logic and Applications (FPL). pp. 1–4. IEEE (2016)
8. Bose, L., Chen, J., Dudek, P.: Descriptor-in-pixel: Point-feature tracking for pixel processor arrays. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5392–5400 (2025)

9. Bose, L., Dudek, P., Carey, S.J., Chen, J.: Live demonstration: Scamp-7. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3995–3996 (2023)
10. Cao, X., Santo, H., Shi, B., Okura, F., Matsushita, Y.: Bilateral normal integration. In: Eur. Conf. Comput. Vis. pp. 552–567. Springer (2022)
11. Cao, X., Shi, B., Okura, F., Matsushita, Y.: Normal integration via inverse plane fitting with minimum point-to-plane distance. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2382–2391 (2021)
12. Carey, S.J., Lopich, A., Barr, D.R., Wang, B., Dudek, P.: A 100,000 fps vision sensor with embedded 535gops/w 256× 256 simd processor array. In: 2013 symposium on VLSI circuits. pp. C182–C183. IEEE (2013)
13. Chatterjee, A., Govindu, V.M.: Efficient and robust large-scale rotation averaging. In: Int. Conf. Comput. Vis. pp. 521–528 (2013)
14. Davison, A.J., Ortiz, J.: Futuremapping 2: Gaussian belief propagation for spatial ai. arXiv preprint arXiv:1910.14139 (2019)
15. Dellaert, F., Kaess, M., et al.: Factor graphs for robot perception. Foundations and Trends® in Robotics **6**(1-2), 1–139 (2017)
16. Fang, W., Zhang, Y., Yu, B., Liu, S.: Fpga-based orb feature extraction for real-time visual slam. In: 2017 International Conference on Field Programmable Technology (ICFPT). pp. 275–278. IEEE (2017)
17. Guan, T., Wang, C., Liu, Y.H.: Neural markov random field for stereo matching. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5459–5469 (2024)
18. Handa, A., Newcombe, R.A., Angeli, A., Davison, A.J.: Real-time camera tracking: When is high frame-rate best? In: Eur. Conf. Comput. Vis. pp. 222–235. Springer (2012)
19. Heep, M., Zell, E.: An adaptive screen-space meshing approach for normal integration. In: Eur. Conf. Comput. Vis. pp. 445–461. Springer (2024)
20. Kavan, L., Collins, S., O'Sullivan, C., Zara, J.: Dual quaternions for rigid transformation blending. Trinity College Dublin **5**, 4 (2006)
21. Kim, H., Jung, Y., Lee, S.: Discontinuity-preserving normal integration with auxiliary edges. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 11915–11923 (2024)
22. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. Adv. Neural Inform. Process. Syst. **24** (2011)
23. Lentaris, G., Stamoulias, I., Soudris, D., Lourakis, M.: Hw/sw codesign and fpga acceleration of visual odometry algorithms for rover navigation on mars. IEEE Transactions on Circuits and Systems for Video Technology **26**(8), 1563–1577 (2015)
24. Li, J., Wang, H., Irshad, M.Z., Vasiljevic, I., Walter, M.R., Guizilini, V.C., Shakhnarovich, G.: Fastmap: Revisiting structure from motion through first-order optimization. arXiv preprint arXiv:2505.04612 (2025)
25. Liu, S., Yu, Y., Pautrat, R., Pollefeys, M., Larsson, V.: 3d line mapping revisited. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21445–21455 (2023)
26. Liu, Y., Bose, L., Fan, R., Dudek, P., Mayol-Cuevas, W.: On-sensor binarized CNN inference with dynamic model swapping in pixel processor arrays. Frontiers in Neuroscience **16** (2022)
27. Ltd, G.: IPU Processors. https://www.graphcore.ai/products/ipu
28. Mandal, D.K., Jandhyala, S., Omer, O.J., Kalsi, G.S., George, B., Neela, G., Rethinagiri, S.K., Subramoney, S., Hacking, L., Radford, J., et al.: Visual inertial odometry at the edge: A hardware-software co-design approach for ultra-low latency and power. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 960–963. IEEE (2019)

29. Mazur, K., Bae, G., Davison, A.J.: Superprimitive: Scene reconstruction at a primitive level. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4979–4989 (2024)
30. Murai, R., Alzugaray, I., Kelly, P.H., Davison, A.J.: Distributed simultaneous localisation and auto-calibration using gaussian belief propagation. IEEE Robot. and Auto. Letters **9**(3), 2136–2143 (2024)
31. Murai, R., Ortiz, J., Saeedi, S., Kelly, P.H., Davison, A.J.: A robot web for distributed many-device localization. IEEE Trans. on Robotics **40**, 121–138 (2023)
32. Murai, R., Saeedi, S., Kelly, P.H.: Bit-vo: Visual odometry at 300 fps using binary features from the focal plane. In: IEEE/RSJ Int. Conf. on Intell. Robots and Systems. pp. 8579–8586. IEEE (2020)
33. Nabarro, S., Van Der Wilk, M., Davison, A.J.: Learning in deep factor graphs with gaussian belief propagation. arXiv preprint arXiv:2311.14649 (2023)
34. Nagata, J., Sekikawa, Y.: Tangentially elongated gaussian belief propagation for event-based incremental optical flow estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21940–21949 (2023)
35. Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: Dtam: Dense tracking and mapping in real-time. In: Int. Conf. Comput. Vis. pp. 2320–2327. IEEE (2011)
36. Nikolic, J., Rehder, J., Burri, M., Gohl, P., Leutenegger, S., Furgale, P.T., Siegwart, R.: A synchronized visual-inertial sensor system with fpga pre-processing for accurate real-time slam. In: IEEE Int. Conf. Robot. Autom. pp. 431–437. IEEE (2014)
37. Ortiz, J., Evans, T., Davison, A.J.: A visual introduction to gaussian belief propagation. arXiv preprint arXiv:2107.02308 (2021)
38. Ortiz, J., Pupilli, M., Leutenegger, S., Davison, A.J.: Bundle adjustment on a graph processor. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2416–2425 (2020)
39. Pan, L., Baráth, D., Pollefeys, M., Schönberger, J.L.: Global structure-from-motion revisited. In: Eur. Conf. Comput. Vis. pp. 58–77. Springer (2024)
40. Pataki, Z., Sarlin, P.E., Schönberger, J.L., Pollefeys, M.: Mp-sfm: Monocular surface priors for robust structure-from-motion. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21891–21901 (2025)
41. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier (2014)
42. Peterson, C., Hartman, E.: Explorations of the mean field theory learning algorithm. Neural Networks **2**(6), 475–494 (1989)
43. Qi, X., Liao, R., Liu, Z., Urtasun, R., Jia, J.: Geonet: Geometric neural network for joint depth and surface normal estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 283–291 (2018)
44. Qi, X., Liu, Z., Liao, R., Torr, P.H., Urtasun, R., Jia, J.: Geonet++: Iterative geometric neural network with edge-aware refinement for joint depth and surface normal estimation. IEEE Trans. Pattern Anal. Mach. Intell. **44**(2), 969–984 (2020)
45. Qin, S., Liu, Q., Yu, B., Liu, S.: π-ba: Bundle adjustment acceleration on embedded fpgas with co-observation optimization. In: 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). pp. 100–108. IEEE (2019)
46. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4104–4113 (2016)
47. Scona, R., Matsuki, H., Davison, A.: From scene flow to visual odometry through local and global regularisation in markov random fields. IEEE Robot. and Auto. Letters **7**(2), 4299–4306 (2022)

48. Shoemake, K.: Animating rotation with quaternion curves. In: Proceedings of the 12th annual conference on Computer graphics and interactive techniques. pp. 245–254 (1985)
49. So, H.M., Bose, L., Dudek, P., Wetzstein, G.: Pixelrnn: in-pixel recurrent neural networks for end-to-end-optimized perception with neural sensors. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 25233–25244 (2024)
50. Sola, J., Deray, J., Atchuthan, D.: A micro lie theory for state estimation in robotics. arXiv preprint arXiv:1812.01537 (2018)
51. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
52. Sucar, E., Liu, S., Ortiz, J., Davison, A.J.: imap: Implicit mapping and positioning in real-time. In: Int. Conf. Comput. Vis. pp. 6229–6238 (2021)
53. Vemulapati, V., Chen, D.: Fslam: an efficient and accurate slam accelerator on soc fpgas. In: 2022 International Conference on Field-Programmable Technology (ICFPT). pp. 1–9. IEEE (2022)
54. Wang, R., Xu, S., Dai, C., Xiang, J., Deng, Y., Tong, X., Yang, J.: Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5261–5271 (2025)
55. Yates, T., Cheng, Y., Alzugaray, I., Akarca, D., Mediano, P.A.M., Davison, A.J.: Belief propagation converges to gaussian distributions in sparsely-connected factor graphs. In: Int. Conf. Mach. Learn. (2026)
56. Yi, B., Lee, M.A., Kloss, A., Martín-Martín, R., Bohg, J.: Differentiable factor graph optimization for learning smoothers. In: IEEE/RSJ Int. Conf. on Intell. Robots and Systems. pp. 1339–1345. IEEE (2021)
57. Yuan, W., Gu, X., Dai, Z., Zhu, S., Tan, P.: Neural window fully-connected crfs for monocular depth estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3916–3925 (2022)
58. Zhang, J.: The mean field theory in em procedures for markov random fields. IEEE Transactions on signal processing **40**(10), 2570–2583 (2002)
59. Zhang, Z., Suleiman, A.A., Carlone, L., Sze, V., Karaman, S.: Visual-inertial odometry on chip: An algorithm-and-hardware co-design approach. In: Robotics: Science and Systems. vol. 13. Robotics: Science and Systems (2017)
60. Zhu, Z., Peng, S., Larsson, V., Cui, Z., Oswald, M.R., Geiger, A., Pollefeys, M.: Nicer-slam: Neural implicit scene encoding for rgb slam. In: 2024 International Conference on 3D Vision (3DV). pp. 42–52. IEEE (2024)
61. Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M.R., Pollefeys, M.: Nice-slam: Neural implicit scalable encoding for slam. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 12786–12796 (2022)
62. Zhuang, B., Cheong, L.F., Lee, G.H.: Baseline desensitizing in translation averaging. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4539–4547 (2018)